International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8, Issue-1, May 2019

An innovative Technique to Protect a Register File from Multiple-bit-upsets Implemented on FPGA.

Ravi Dontaraju, S. Bhujanga Rao.

Abstract*: *The development of electronic devices which possess a characteristic to resist radiation-induced upsets is an active area of research in the semiconductor industry. There are several techniques beginning from the manufacturing level of the transistors to the implementation level of a chip, which helps in masking or eliminating faults caused by radiation. In this paper, an implementation level technique is proposed which can mask multiple bit-upsets caused by radiation in a register file implemented on an FPGA. The advantage of this technique is that masking of the errors is done by leveraging the inherent resources present in an FPGA due to which there shall be a significant decrease in the area overhead without a compromise in the reliability and delay of the system.

Index Terms*: *Faults, FPGA, Radiation-induced upsets, Reliability, Semi-Conductor.

I. INTRODUCTION

The electronic devices which are deployed in the space and avionic applications should have high accuracy, speed and yet the main characteristic which they have to possess is the robustness towards the radiation-induced upsets. Modern-day Static-RAM (SRAM)-based Field Programmable Gate Arrays (FPGAs) have very high importance in space and avionics applications due to their high-operation limit, performance and most importantly dynamic reconfigurability before or after deployment [1].

Though the FPGA's are suitable for space applications they do not completely oppose the radiation-induced faults, they can be made radiation-proof by using some extra hardware which is known as radiation hardening [2].

One of the most prime components to consider is a register file as it stores the intermediate products of computation and critical program flow execution information. When the data in it gets corrupted it will result in silent data corruption (SDC), a wrong branch address or an output failure [3]. Hence, it must be critically designed to withstand the radiation upsets and provide an error-free output.

Revised Manuscript Received on 30 May 2019.

***** Correspondence Author

Ravi Dontaraju*, Department of Electronics and Communication Engineering, Sreenidhi institute of Science and technology, Hyderabad, India.

S. Bhujanga Rao, Professor, Director-SOE, Department of Electronics and Communication Engineering, Sreenidhi institute of Science and technology, Hyderabad, India.

In this paper, we propose a technique to protect the register file from multiple-bit-upsets by leveraging the inherent resources present in FPGA.

II. BACKGROUND WORK

Space and avionic applications are considered to be the backbone for the development of any nation it facilitates international communication and as well as maintenance of world peace by satellite surveillance 24x7. Yet to build such systems they have to satisfy a basic characteristic of robustness towards the effects of radiation.

The faults caused by radiation can be categorized into two types Total Ionization Dose (TID) and Single Event Effects (SEE).

Total Ionizing Dose is a complex effect. It causes defects in the semiconductor structure itself and decreases the mobility of the charge carriers. The advancements in the process technology like epitaxial CMOS processes, cell layout and SOI (silicon-on-insulator) have shown promising results in reducing TID [4].

SEE can be categorized into two kinds: hard faults and soft faults. Hard faults include SEL (Single Event Latch-Up), SEB (Single Event Burnout) and SEGR. They cause permanent faults and could be settled mostly by the manufacturing process. Soft faults include SEU (Single Event Upset), MBU Multiple-Bit Upset) and SET (Single Event Transient). They are transient faults which can be mitigated or corrected by design strategies [5].

In this paper we select these SEE as 80% of the radiation leads to SEU's and which in-turn cause MBU's as analyzed in [4] which is a survey of faults occurring in an Algerian satellite in the Low Earth's orbit.

The most sensitive component to soft errors involved in an electronic device is a register file. The occurrence of data error generation due to SEUs is high in a register file [6]-[7]. Typically, a register file is an array of registers where the writing and reading of the data can be particularly selected from the array by mentioning the read and write addresses. The significance of this is seen in digital systems where there are huge calculations which generate intermediate results or also known as temporal contents and these contents are again taken to proceed further, even in multi-threading processors the sequence of thread access is stored in this file itself.

The register file considered is a two-port asynchronous read and synchronous write which is shown in Fig. 1

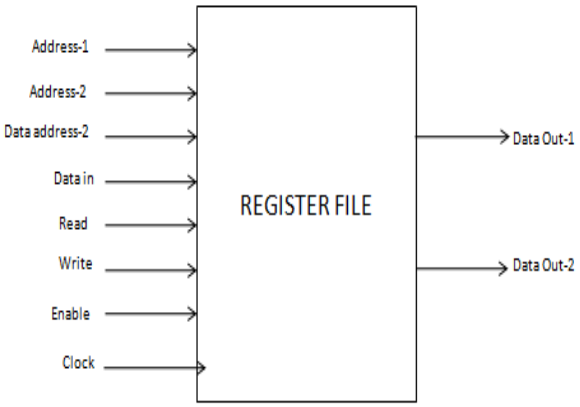

Fig. 1.

III. PREVIOUS WORK

The soft errors caused by radiation are typically bit flips, to detect and correct/mask them there are techniques which can be categorized mainly into two categories namely Design level techniques and Implementation level techniques. The design level techniques make use of EDAC codes such as a technique used in [8] where an extended Hamming (8,4) code is used along with narrow width values which has an advantage of high reliability with less area while having high latency in detection and correction.

Yet in [9] and [10] the detection of the multiple faults is done using parity computation whereas when correction is considered in [9] hamming code is used in diagonal direction increasing the fault coverage and reliability whereas in [10] decimal matrix code is employed with encoder reuse technique which fails to cover large faults but the latency for fault detection and correction is largely scaled down.

An old but powerful technique for multiple errors detection and correction is orthogonal Latin squares proposed in [11] which are worth mentioning as it can mitigate SEU as well as SEFI but having high delay overhead.

The modern-day design level approaches use a redundant circuitry for the detection and correction of errors. In most of the cases, the error detection is done using parity checking as it allows for the detection of a soft error for a minimal cost in terms of circuit complexity and memory width [12]. For instance [13] and [14] consider their detection technique as a multi-directional parity comparison and correction by the concept of erasure coding where copies of the data are stored in a different location and retrieved back when a mismatch occurs. Though [13] and [14] use almost the same techniques the main difference can be pointed is in [13] the authors use the mutation and crossover where there will be a threshold limit beyond which the data is replaced with redundant data otherwise correction is attempted which reduces the computation time whereas in [14] the computation time is less whereas the fault coverage is not that wide. Implementation level techniques are mostly applied on storage resources of a particular platform so as to use the host resources itself as their redundant blocks such as the authors in [15] make an attempt to minimize the area and power consumption where the register file's parity checks are performed and stored in the cache memory, if any mismatch occurs then a CRC is used to correct the errors. The disadvantage faced in this technique is that the processor stalled until the register file is made free from errors.

In [16] the whole register file is duplicated, and parity comparison is done when a mismatch occurs register renaming is done, and errors are masked. The drawback found in this technique is that the applicability of this technique is done only when the whole register file is filled.

The authors in [17] propose a technique to protect the memory elements from single bit errors by using the capabilities of the Xilinx-FPGA Look Up Tables and the architecture of the SLICE-M combined with Error correction codes. The benefits compared with TMR come at a delay cost.

The authors of [18]-[19] make use of a conventional technique known as memory scrubbing to protect FPGA memories, especially the smaller and faster memories such as SRL's(Shift Registers) and LUTRAM's.

The authors of [20] present a logic which incurs delay and resource overhead which is a common frequently observed problem when ECC's are used to protect multi-port memory structures mapped to BRAMs. This can be achieved by exploiting the inbuilt replication of the register file to support multi-port reads in a Xilinx-FPGA.

IV. DESIGN METHODOLOGY

When a register file is mapped onto an FPGA the design tools replicate the register file multiple time in order to realize multi-read ports. This inbuilt redundancy is being leveraged and combined with a good error detection and masking scheme to provide TMR like performance at the reasonably low area and delay overheads. The following sub-sections illustrate the proposed technique in detail.

A. IMPLEMENTATION OF REGISTER FILE ON XILINX-FPGAs.

A Register file is a basic component of a processor-based system and is fundamental in assisting the operations working on numerous registers at the same time. Normally, most tasks require two read ports as information sources. In this way, synchronous perusing of quantities from the register file is a significant thought.

The multi-read port registers files are realized in an SRAM-based FPGAs by the design automation tools which replicate the contents of the register file for more than one instance [21].

In Fig.2, the logical view illustrates the register file before implementation and the physical view illustrates the same register file when it is implemented on an FPGA.

It has to be mentioned that from Fig.2 the write port is connected to both the register files, it becomes synchronous write and an asynchronous read dual port register file.

B. FAULT MASKING BASED ON READ-PORT ADDRESS SWITCHING.

As stated above, the Xilinx design-automation tool replicates the register file in more than one instance to facilitate multiple read ports which are Data Out-1 and Data Out-2 having all the values of the registers (See Fig. 3).

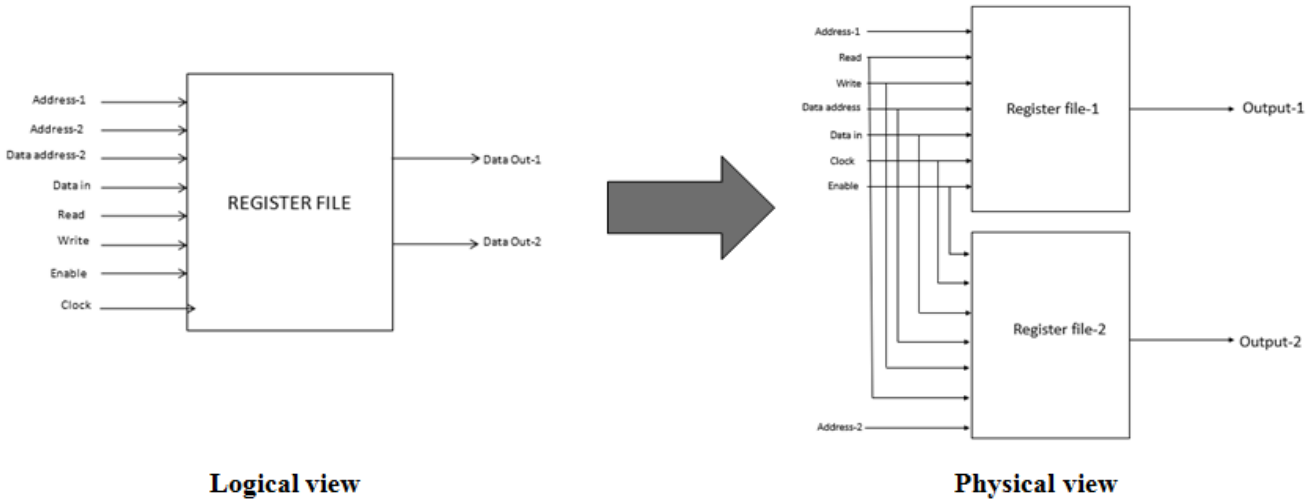

Fig. 2.

This subsection contains a general overview of the proposed technique such that the readers can get the approach overview and can move to further sections. If two registers are read by instruction and let us consider one of the two registers suffers from upset which can be an SEU or MBU's then there will be a change in the horizontal and vertical parities which trigger the swapping circuitry which swaps the read lines, hence masking the upsets. As we consider that the errors occurring pertain only to a single register of a register file there always will be a valid error-free copy from which the data will be accessed.

A demonstration of the technique is shown as an example in Fig. 3. On the left, register r2 in address 1(the actual value is 255) has been corrupted by MBUs at the third, fourth and fifth bits changing its contents to 199. This is reflected at the read operation by the horizontal and vertical parity-check mismatch, but its alternate copy of Address-2 is free of errors and is being used for flipping from the faulty copy.

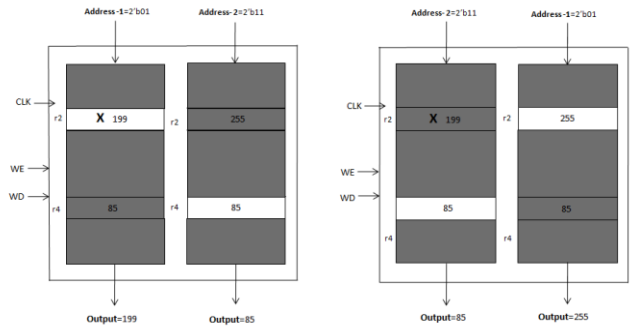

Fig. 3.

An error detection logic is introduced which will be discussed in detail in the next section which detects faults and the switching of the input/output lines. The parity mismatch triggers the swapping circuitry and swaps the input read lines as sown in the right side of Fig. 3. In the example, the fault masking of the technique can be seen at the r2 register which was read from the first copy and when an error is been detected it was read from the alternate copy, hence avoiding the corrupted data.

A similar situation can be observed at the r4 which before was read from Address-2, but after the detection of the fault, it was read from Address-1. In this case, the output of r4 is still valid since none of its copies have suffered an error. A point can be noted that the procedural faults are masked but not corrected. The above-presented issue of procedural faults can be neglected because in most of the cases the register file will be frequently updated and the faulty values are replaced with newer ones.

The next sub-section deals with the implementation of this scheme in Xilinx-FPGA's and a detailed discussion of the implemented scheme.

C. PROPOSED PROTECTION TECHNIQUE

The proposed protection technique for a register file is shown in Fig. 4. It can be observed that its implementation is modified, which includes the parity comparison and storage modules along with several multiplexers and sequential elements. The compactness of the technique can be seen in Fig. 4 where the Look-Up Tables of the memory block are connected directly to the input and output ports. Therefore, the present circuitry connections can be used for parity computation, storage, and retrieval in a synchronous manner with limited overhead.

It can be observed from the figure that the registers R1, R3, and R2, R4 correspond to horizontal and vertical parities of the register file. A point here to be noted is that parity is computed in two directions because parity fails when there are even number of errors in the memory and this can be tackled by using two-dimensional parity.

When the register file is loaded with values by enabling the write pin the horizontal and vertical parities are computed and stored in R1 and R2 registers.

The registers R3 and R4 are loaded with the horizontal and vertical parity values instantly when the Read signal gets high. The registers holding the horizontal values and the registers holding the vertical parity values of the register file are compared with each other and the comparison output is sent to O1 which raises a parity-check flag which is serially connected to a flip-flop DFF-2.

The flip-flop samples the parity generated signal and also breaks the combinational elements feedback cycle. Generally, two cases are dealt with our proposed protection technique which are, the operation of a command designed to deal with two separate operands (like ADD R1, R2, R3) and when a command is using the same register (like OR R1, R2). Hence Address-1 and Address-2 lines are compared to check whether same registers are accessed by both the read lines and the value serves as a selection input to multiplexers M5 and M6 which deal with the above stated two cases. In the case of un-common registers selection, the same-signal operator will result in a logic-0 which directs to DFF-2's output. In the case of a common register selection, the same-signal operator will result in a logic-1 directing the outputs from DFF-1 and DFF-3 to be selected. A point is to be noted that these flip-flops straight-away sample the in-equality operators reflecting the copy which has suffered a fault and sequentially will flip the input and output lines just by using M1 and M2 multiplexers at the input stage and M3 and M4at the output stage of the memory.

From the analysis of the above two cases, we can say that the faulty copy of memory is completely masked, and data is accessed from the alternate replicated copy.

The comparison of the operating cycle with the delay overhead, the delay generated due to the error detection and masking logic is nearly negligible.

This technique doesn't require the stalling of processor pipeline stages for long as the mechanism of the entire technique ends before the next rising edge of the clock cycle, thus providing a valid and required data to the successor stages of the execution flow.

V. EVALUATION

The proposed technique is evaluated by considering two register files and specifying a TEST input which when is logic-0 the write data is stored in both the registers synchronously which is similar to the FPGA replication to realize multi-read port register file and when the TEST is logic-1 the data is loaded only in the first register leaving the second register with the previously loaded data. This indirectly looks like there has been a fault occurred and as per our proposed technique, the outputs are evaluated.

VI. SIMULATION RESULTS AND DISCUSSION

As per the evaluation procedure as discussed in the above section various faults which have been injected into the register file as inputs and can be segregated into three types: A Single Event Upset, Multiple Even Upsets and Multiple Odd upsets.

To evaluate the register file given while TEST=0 are 85,170, 255, 129 considering a register file of 8-bit width and four registers depth.

For Single event upset the fifth bit of the third register is flipped changing the value from 255 to 239, the output simulation results can be seen in
Fig. 5.

For multiple even bit upsets the third, fourth, fifth and sixth

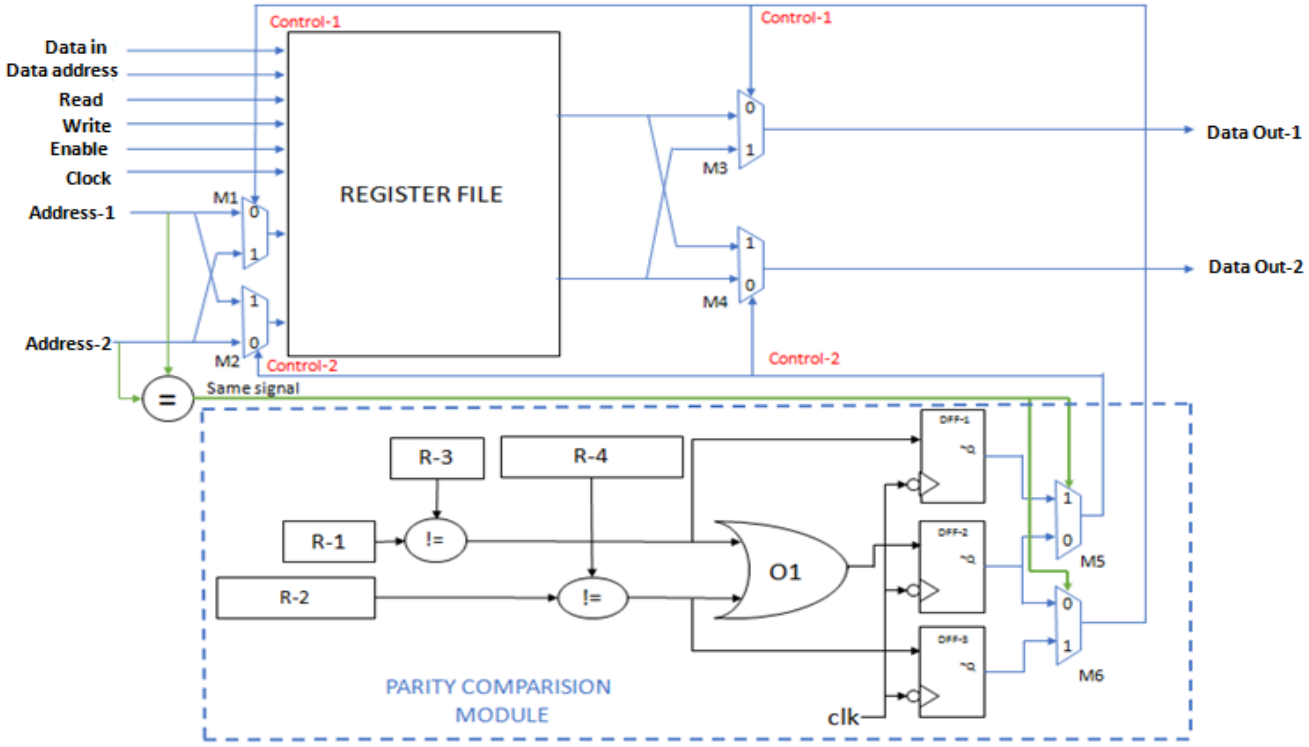

Fig. 4.

bits of the third register of the register file are flipped, changing the value for 255 to 195. The results can be viewed in Fig. 7 in the form of simulation waveforms.

For multiple odd bit upsets the fourth, fifth and sixth bits of the third register of the register file are flipped, changing the value for 255 to 199. The results can be viewed in Fig. 7 in the form of simulation waveforms.

The behavior of the proposed technique can be clearly observed in the simulated waveforms shown in Fig. 5, Fig. 6 and Fig. 7. The number of resources the design is consuming is also an important aspect to be considered in the semiconductor industry. The implementation results of the technique are compared with the traditional TMR technique which is shown in Table 1.

	Register file		Masking resources		
	LUTRAMs	Slice M	LUTs	FF	SliceL
Default	88	22	0	0	0
TMR	264	66	128	0	43
Proposed	88	22	129	3	44

Table. 1.

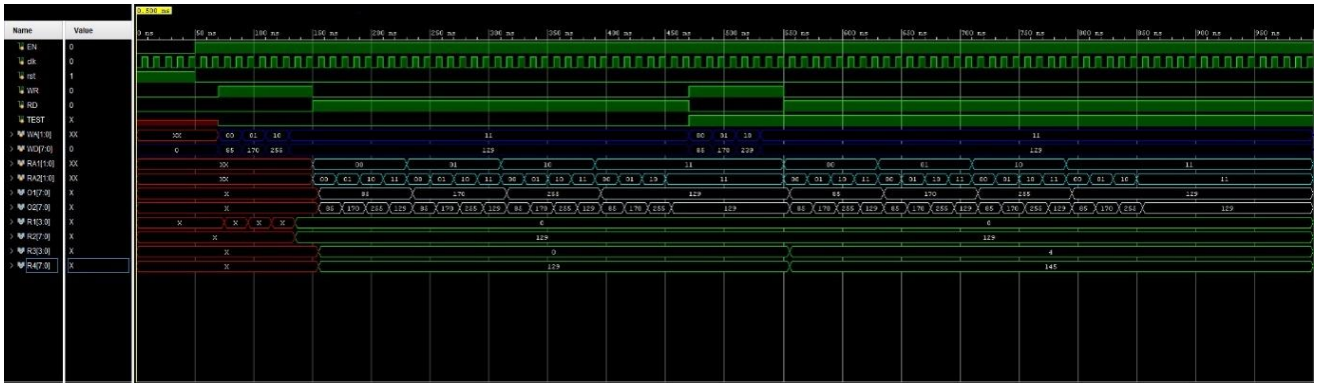

Fig. 5.

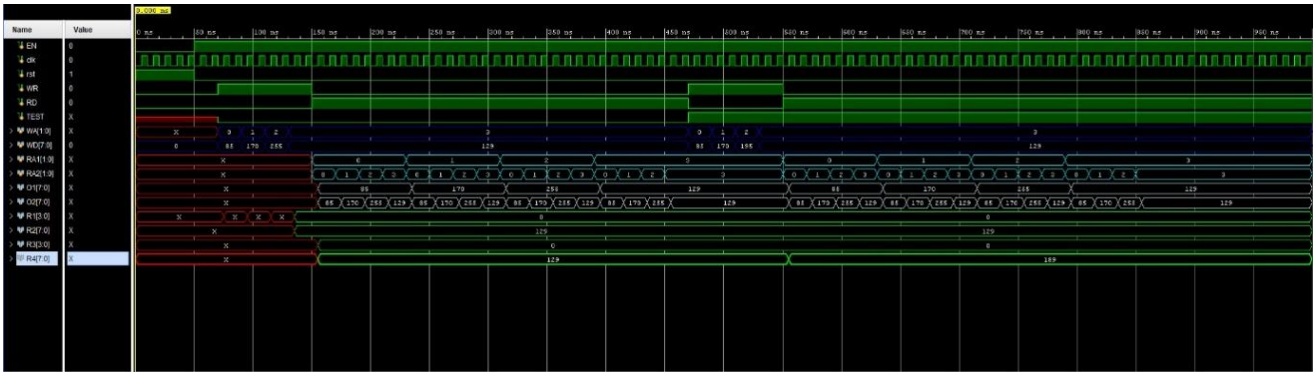

Fig. 6.

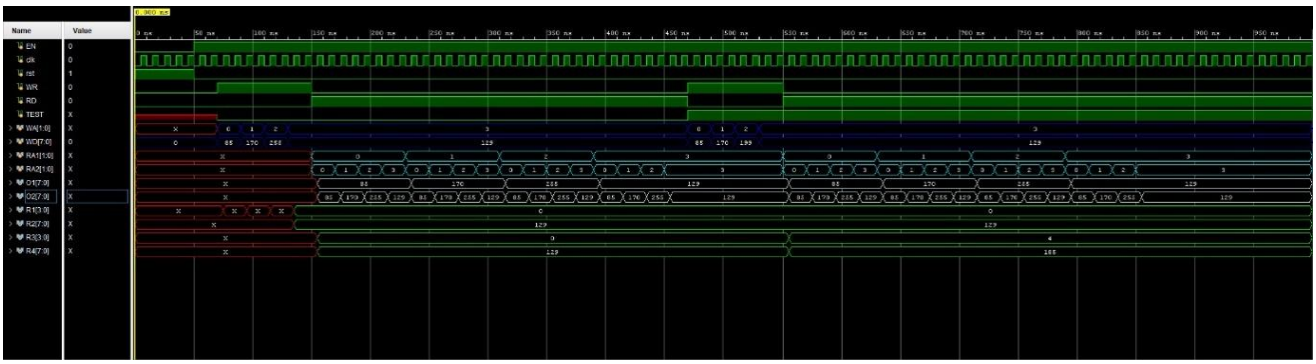

Fig. 7.

I. CONCLUSIONS AND FUTURE WORK

This paper presents a innovative technique to protect the register file implemented in Xilinx SRAM-based FPGA. The method is built around the inbuilt redundancy resources given by the design automation tools of Xilinx-FPGA which help in generating a low overhead in terms of area and delay for the detection and masking of errors than TMR.

Based on the demonstrated results our technique is applicable to a register file and also can be extended to other higher architecture memory structures which when implemented through distributed RAM resources on SRAM-based FPGAs. The proposed two-dimensional parity-based error detection is able to mask multiple bit errors occurring in a single register of the register file. Future work will be considered if multiple bit errors are corrected instead of masking as few applications do not frequently update the register file and it becomes a troublesome issue when the fault stays longer.

REFERENCES

1. Ratter, *FPGAs on Mars*, ser. Journal n50. Xilinx Technical Report, Cell Jornal, 2004.
2. 2. M. Swift "Xilinx Single Event Effects I" Consortium Report Virtex-II Static SEU Characterization" Tech.rep., Jet Propulsion Laboratory, and California Institute of Technology, Jan 2004.
3. Bairavasundaram, L.N. Arpaci-Duseau, R. H., Goodson, G. R., and Schroeder, B., "An analysis of data corruption in the storage stack, " Nov. 2008.
4. Y.Bentoutou, M.Djaifri, P.Sellin, "Observations of single-event upsets and multiple-bit upsets in random access memories onboard the Algerian satellite", Oct. 2008
5. Shangqing Zhang and Hongjin Liu, "Synthetical analysis on space radiation tolerance techniques in ASICs and FPGAs," *2011 International Conference on System science, Engineering design, and Manufacturing informatization*, Guiyang, 2011, pp. 305-310. doi: 10.1109/ICSSEM.2011.6081305
6. G.P. Saggese, N.J.Wang, Z.T.Kalbarzyk, and R.K.Iyer, "An Experimental Study of Soft Errors in Microprocessors", Nov. 2005.
7. G. Memk, M.T. Kander, and O. Ozturk "Increasing Register File Immunity to Transient Errors", Mar. 2005.
8. Siamak Esmaeli, Mortza. Hosseini, Bijan Vosouhi. Vahdat, Bizhan. Rashidin, "A multi-bit error tolerant register file for a highly reliable embedded processor", 2011 18th IEEE International Conference on Electronics Circuits and Systems, pp. 532-537.
9. S. Venkataraman, R. Santos, S. Maheshwari, A. Kumar, "Multi-directional error correction schemes for SRAM-based FPGAs", 2014.
10. Lakshmeepathiputtappa KB and Dr. Sarika Tale. "FPGA Implementation of Error Detection and Correction using Decimal Matrix Code." *International Journal for Innovative Research in Science & Technology* 2, no. 4 (2015): 1-9.
11. A. Cóbreces, J. Tbero, A. Regdío, A. Sánchez-Macán, P. Reviregо, and
12. J. A. Mastro, "SEU and SEFI Protection for DDR3 Memories in a Xilinx Zynq-7000 FPGA", 2017.
13. R. Baumann, "Soft errors in advanced computer systems," in IEEE Design & Test of Computers, vol. 22, no. 3, pp. 258-266, May-June 2005.
14. S. Aishwarya, G. Mahendran, "Multiple bit upset corrections in SRAM based FPGA using Mutation and Erasure codes", *2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pp. 202-206, 2016.
15. M. Ebrahimi, P.M.B. Rao, R. Seyyedi, M.B. Tahoori, "Low-cost multiple bit upset corrections in SRAM-based FPGA configuration frames", *Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 3, pp. 932-943, 2016.
16. J. A. Blome, S. Gupta, S. Feng, and S. Mahlke, "Cost-efficient soft error protection for embedded microprocessors," in Proceedings of the 2006 international conference on Compilers, architecture, and synthesis for embedded systems. ACM, 2006, pp. 421–431.
17. G. Memik, M. T. Kandemir, and O. Ozturk, "Increasing register file Immunity to transient errors," in Design, Automation, and Test in Europe, March 2005, pp. 586–591 Vol. 1.
18. Reviriego, P., Demirci, M., Tabero, J., Regadío, A., and Maestro, J. A., "DMR +: An efficient alternative to TMR to protect registers in Xilinx FPGAs," Microelectronics Reliability, vol. 63, pp. 314 – 318, 2016.
19. P. Adell, G. Allen, G. Swift, and S. McClure., "Assessing and mitigating radiation effects in Xilinx SRAM FPGAs," in 2008 European Conference on Radiation and Its Effects on Components and Systems, Sept 2008, pp. 418–424.
20. N. H. Rollins, "Hardware and software fault-tolerance of softcore processors implemented in SRAM-based FPGAs," Doctoral Dissertation, BYU 2012.
21. C. Hong, K. Benkrid, X. Iturbe and A. Ebrahim., "Design and implementation of fault-tolerant soft processors on FPGAs," in 22nd International Conference on Field Programmable Logic and Applications (FPL), Aug 2012, pp. 683–686.
22. Xilinx, XST User Guide for Virtex-4, Virtex-5, Spartan-3, and Newer CPLD Devices, http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_4/xst.pdf, accessed at December 2013.

AUTHORS PROFILE

Ravi Dontaraju received his bachelor's degree in Electronics and Communication Engineering form Sreyas Institute of Engineering and Technology, Hyderabad, India in the year 2017. Presently pursuing masters in digital systems and Computer Electronics from Sreenidhi Institute of Science and Technology Hyderabad, India. His areas of interest include front end VLSI design and fault testing of digital systems.

Prof. S. Bhujanaga Rao graduated in Electronics Engineering from Madras Institute of Technology in 1974. He joined Electronics Corporation of India Limited (ECIL), started as an engineer and later took up research in consumer electronics products such as TV transmitters and receivers, Projection TVs, etc. While working at ECIL he completed his post-graduate program in Advanced Electronics from JNTU Hyderabad, 1984. After 25 years of service in the industry, he took retirement to fulfill his passion for teaching. Prof. Rao joined the ECE department at SNIST in the year 2000 as an associate professor. He has been actively teaching and working ever since and currently he is the Director, School of Electronics and Dean of Industry Institute Interaction at SNIST.